Self Introduction and RD53A Project

Nikola Whallon

University of Washington

December 2, 2016, LBNL Instrumentation Meeting

Introduction

- I'm Nikola Whallon, a graduate student at the University of Washington, Shih-Chieh Hsu is my supervisor
- I'll work at LBNL for 1 year under Maurice on RD53A
- LBNL should receive the RD53A chip in 2017, and I will help to prepare the test setup to benchmark and characterize it
- I'll work closely with Timon since we will use the YARR test-stand software
- I have experience working on pixel software - PixLib, STcontrol, Calibration Console

FEI4/RD53A Software Emulator

Since the RD53A chip may not arrive for a while (summer 2017), I will begin by writing a simple software emulator for the chip to help further test and develop YARR. The bare-bones of the emulator must:

- receive commands from YARR (we will use shared memory)
- interpret and execute the commands, like wrRegister, etc (we will use a state machine to manage this)
- send data back to YARR (we will use shared memory)

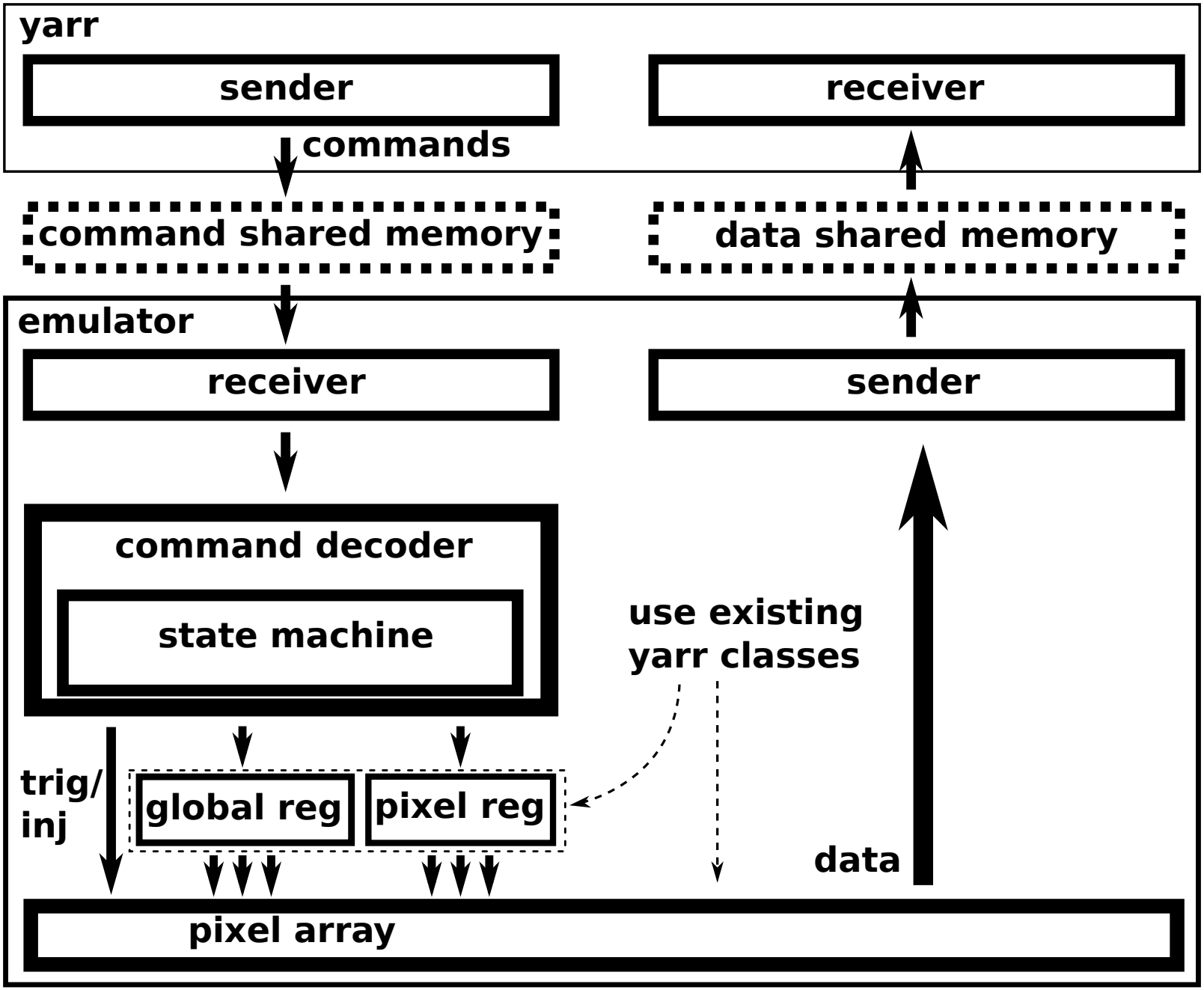
I have begun some implementation of this emulator, and working on it helps me ramp up my familiarity with YARR.

Some Emulator Functionality

For the emulator to be helping in developing YARR, it must provide the functionality to perform scans, tunes, etc. A non-inclusive list of features is:

- implement a pixel array
- allow triggers and injection
- model ToT as a function of charge
- model threshold as a function of charge, global threshold, and TDAC values
- handle digital scans, analog scans, threshold scans, GDAC tunes, TDAC tunes

Emulator Flow Chart



wrRegister (Some Details)

This week, I wrote a basic emulator skeleton following the design from the previous slide. YARR sends wrRegister commands in the following format:

```
core->writeFifo(0x005A0800+((chipId<<6)&0x3C0)+(address&0x3F));
core->writeFifo((value<<16)&0xFFFF0000);
```

I have a dummy program writing this into shared memory, and then the emulator can grab the command from the shared memory:

```
memcpy(&command, &shm_command_pointer[0], 4);
memcpy(&value, &shm_command_pointer[4], 4);
```

The bits then get decoded to give me an `address`, `value`, and `chipID`. I can then simply write the `value` into the proper `address` of a YARR Fei4 object:

```
fe->cfg[address] = value;
```

This shows how using existing YARR classes can be very useful!

Summary

- a software emulator of FE chips will greatly help in the development and testing of YARR
- this will prepare us to quickly begin testing RD53A when it arrives at LBNL in 2017
- since the emulator is software, it is great for quick developing and beginners, as no hardware is necessary
- development of the emulator has begun and functionality is being added
- I look forward to working at LBNL!